

TD N°3 : Les algorithmes de recherche et de tri (Corrigé)

Exercices 1 : Gestion des notes d'une classe

On considère un tableau de taille $N_{max} = 100$. Ce tableau contient les notes des étudiants. Écrivez un algorithme où vous déclarez ce tableau, demandez un entier N qui représente le nombre des étudiants puis réalisez les traitements suivants :

1. Ecrire une procédure qui calcule et affiche la moyenne des notes.
2. Ecrire une procédure qui calcule et affiche la note minimale.
3. Ecrire une procédure qui calcule et affiche la note maximale
4. Ecrire une procédure qui cherche si une note donnée existe dans le tableau ou pas et affiche le nombre d'occurrence ;
5. Ecrire une procédure qui modifie une note donnée du tableau T connaissant son indice, et affiche le tableau après modification ;
6. Ecrire une procédure qui inverse l'ordre des éléments d'un tableau préalablement saisi
7. Ecrire le programme principal qui saisit les N notes dans un tableau, et invoque les différentes procédures.

Solution :

1. Ecrire une procédure qui calcule et affiche la moyenne des notes.

```
Procédure Moy_Tableau (T[] : Réel, N : Entier)
  Variables i : Entier
           S, Moy : Réel

  Début
    Pour i ← 0 à N-1 faire
      S ← S+T[i]
    FinPour
    Moy ← S/N ;
    Ecrire("la moyenne des notes est Moy =", Moy)
  Fin
```

```
# Ecrire une procédure qui calcule et affiche la moyenne des notes.
def Moy_Tableau (T, N):
    S=0
    for i in range(0,N):
        S=S+T[i]
    Moy=S/N
    print("La moyenne des notes est Moy =", Moy)
```

2. Ecrire une procédure qui calcule et affiche la note minimale.

```
Procédure Min_Tableau (T[] : Réel, N : Entier)
  Variables i : Entier
           Min : Réel
  Début
    Min ← T[0]
    Pour i ← 1 à N-1 faire
      Si(T[i]<Min) alors
        Min ← T[i]
      Finsi
    FinPour
    Ecrire ("la note minimale est ", Min)
  Fin
```

```
# Ecrire une procédure qui calcule et affiche la note minimale.
def Min_Tableau (T: float, N : int) :
  Min = T[0]
  for i in range (1, N) :
    if (T[i]<Min) :
      Min=T[i]
  print("La note minimale est ", Min)
```

3. Ecrire une procédure qui calcule et affiche la note maximale.

```
Procédure Max_Tableau (T[] : Réel, N : Entier)
  Variables i : Entier
           Max : Réel
  Début
    Max ← T[0]
    Pour i ← 1 à N-1 faire
      Si(T[i]>Max) alors
        Max ← T[i]
      Finsi
    FinPour
    Ecrire ("la note maximale est ", Max)
  Fin
```

```
# Ecrire une procédure qui calcule et affiche la note maximale.
def Max_Tableau (T : float, N : int) :
  Max = T[0]
  for i in range(1, N):
    if (T[i]>Max) :
      Max=T[i]
  print ("La note maximale est ", Max)
```

4. Ecrire une fonction qui cherche si une note donnée existe dans le tableau ou pas et retourne le nombre d'occurrence ;

Procédure Occurrence_Tableau (T[] : Réel, N : Entier)

Variables i, c : Entier
X, a : Réel

Début

Ecrire("donner l'élément que vous voulez chercher dans le tableau ")

Lire(X);

a←0

c←0

Pour i ←0 à N-1 faire

Si(T[i]=X)

a←1

c←c+1

Finsi

FinPour

Si(a=1)

Ecrire("la note" ,X, " se trouve dans le tableau ")

Ecrire("le nombre d'occurrence de la note" ,X, " est =",c)

Sinon

Ecrire("la note ",X," ne se trouve pas dans le tableau")

FinSi

Fin

```
#. Ecrire une fonction qui cherche si une note donnée existe dans le tableau ou pas et retourne  
def Occurrence_Tableau (T: float, N : int):  
    print("donner l'élément que vous voulez chercher dans le tableau ")  
    X=int(input())  
    a=0  
    c=0  
    for i in range(0,N):  
        if (T[i]==X):  
            a=1  
            c=c+1  
    if(a==1):  
        print("La note" ,X, " se trouve dans le tableau ")  
        print("Le nombre d'occurrence de la note" ,X, " est =",c)  
    else:  
        print("La note ",X," ne se trouve pas dans le tableau")
```

5. Ecrire une procédure qui modifie une note donnée du tableau T connaissant son indice, et afficher le tableau après modification ;

Procédure Modification_Tableau (T[] : Réel, N : Entier)

Variables i, p : Entier
X : Réel

Début

Ecrire("Entrer l'indice de l'élément à modifier ")

Lire (p)

Si (p<0) ou (p>=N) **alors**

Ecrire("Position hors limites du tableau ")

Sinon

Ecrire("L'ancienne valeur dans cette position est : " , T[p])

Ecrire("Entrer la nouvelle valeur: ")

```

        Lire(X)
        T[p] ← X
    Finsi
    // affichage des éléments du tableau après la modification
    Pour i←0 à N-1 faire
        Ecrire (T[i])
    FinPour
Fin

```

```

# Ecrire une procédure qui modifie une note donnée du tableau T connaissant son indice, et affiche
def Modification_Tableau (T: float, N : int) :
    print("Entrer l'indice de l'élément à modifier ")
    p=int(input())
    if(p<0) or (p>=N):
        print("Position hors limites du tableau")
    else:
        print("L'ancienne valeur dans cette position est : " , T[p])
        print("Entrer la nouvelle valeur: " )
        X=int(input())
        T[p]=X
    #affichage des éléments du tableau après la modification
    for i in range(0, N):
        print(T[i])

```

6. Ecrire une procédure qui inverse l'ordre des éléments d'un tableau préalablement saisi

```

Procédure Inverse_ordre (T[] : Réel, N : Entier)
Variables i, Temp : Entier
Début
    Pour i ← 0 à (N-1)/2 faire
        Temp ← T(i)
        T(i) ← T(N-1-i)
        T(N-1-i) ← Temp
    FinPour
    // affichage des éléments du tableau après la modification
    Pour i←0 à N-1 faire
        Ecrire (T[i])
    FinPour
Fin

```

```

# Ecrire une procédure qui inverse l'ordre des éléments d'un tableau préalablement saisi
def Inverse_ordre (T: float, N : int):
    for i in range(0, N//2):
        Temp = T[i]
        T [i] = T[N-1-i]
        T[N-1-i] = Temp

    # affichage des éléments du tableau après la modification
    for i in range (0, N):
        print(T[i])

```

7. Ecrire le programme principal qui saisit les N notes dans un tableau, et invoque les différentes procédures.

```

Algorithme notesEtudiants
Variables
    Tableau T[1...100] : réel
Début
Répéter

```

```

Ecrire ("Entrer le nombre de notes à saisir")
Lire(N)
Jusqu'à (N>=0 ET N<100)
Pour i ← 1 à N faire
    Ecrire ("Entrer la note N° ", i)
    Lire(T[i])
FinPour
Moy_Tableau (T[], N)
Min_Tableau (T[],N)
Max_Tableau (T[], N)
Occurrence_Tableau (T[], N)
Modification_Tableau (T[], N)
Inverse_ordre (T[], N)

```

```

#Programme principal

print ("Entrer le nombre de notes à saisir")
N=int(input())

while (N<=0 and N>100):
    print ("Entrer le nombre de notes à saisir")
    N=int(input())

#T=array("f", )
T=[]
for i in range (0, N):
    print("Entrer la note N° ", i)
    x=float(input())
    T.append(x)

Moy_Tableau (T, N)
Min_Tableau (T,N)
Max_Tableau (T, N)
Occurrence_Tableau (T, N)
Modification_Tableau (T, N)
Inverse_ordre (T, N)

```

Exercice 2 : tri d'un tableau (par insertion)

Ecrire une procédure qui effectue le tri d'un tableau envoyé en argument (on considère que le programme appelant devra également fournir le nombre d'éléments du tableau).

```

Procédure TriTableau(T[], n :Entier)
    Variables i, j, tmp : Entier
    Début
        Pour i ← 1 à n-1 faire
            tmp ← T[i]
            Tantque (j>=0 ET T[j]>tmp) faire
                T[j+1] ← T[j]
                j ← j-1
            Fintantque
            T[j+1] ← tmp
        FinPour
    Fin

```

```

"""Ecrire une procédure qui effectue Le tri d'un tableau envoyé en argument
(on considère que Le programme appelant devra également fournir Le nombre d'éléments du
tableau)
"""
def TriTableau(T, n) :
    for i in range(1, n) :
        tmp=T[i]
        j=i-1
        while(j>=0 and T[j]>tmp) :
            T[j+1]=T[j]
            j=j-1
        T[j+1]=tmp

# Version 1 : Tableau déjà rempli
T=[100,22,78,54,12,54,3,24]
print("Avant Le Tri", T)
TriTableau(T, len(T))
print("Après Le Tri", T)
# Version 2 : éléments du tableau à saisir
N=int(input("Donner La taille du Tableau: "))
T=[] #T=[0]*N
for i in range(0,N):
    T.append(int(input(f"Element T[{i}]")))
    #T[i]=int(input(f"Element T[{i}]"))
print("Avant Le Tri", T)
TriTableau(T, len(T))
print("Après Le Tri", T)

```

Exercice 3 : Tableau trié ou non

Ecrire une procédure qui informe si un tableau envoyé en argument est formé d'éléments tous rangés en ordre croissant ou non.

<p>Variable Flag : Booléen Procédure TableauCroissant(T[] , n : Entier) Variable i : Entier Début Flag ← Vrai i ← 0 TantQue(Flag ET i < n-1) faire Flag ← T[i] < T[i+1] i ← i+1 FinTantQue Fin</p>

```

"""
Ecrire une procédure qui informe si un tableau envoyé en argument est formé d'éléments
tous rangés en ordre croissant ou non.
"""
Flag=True
def TableauCroissant(T, n) :
    global Flag
    i=0
    while(Flag and i<n-1) :
        Flag=T[i]<=T[i+1]
        i=i+1

T=[14,3, 1, 25, 24, 55, 54, 78, 100]
print(len(T))
print(T)
TableauCroissant(T, len(T))

if(Flag==True):
    print("Les éléments du tableau sont en ordre croissant")
else:
    print("Les éléments du tableau ne sont pas en ordre croissant")

```

Exercice 4 : Tri croissant ou décroissant

Ecrire une procédure qui effectue le tri d'un tableau (La procédure prendra un paramètre booléen: Vrai - le tri devra être effectué dans l'ordre croissant, FAUX - dans l'ordre décroissant).

```

Procédure TriTableau(F : Booléen, T[] , n : Entier)
Variables i, pos, temp : Entier
Début
    Pour i ← 0 à n-2 faire
        pos ← i
        Pour j ← i + 1 à n-1 faire
            Si F Alors //F est Vrai => ordre croissant
                Si (t[j] < t[pos]) Alors
                    pos ← j
                Finsi
            Sinon
                Si (t[j] > t[pos]) Alors
                    pos ← j
                Finsi
        Finsi
        Finpour
        temp ← T[pos]
        T[pos] ← T[i]
        T[i] ← temp
    FinPour
Fin

```

```

"""
Ecrire une procédure qui effectue le tri d'un tableau (La procédure prendra un paramètre
booléen: Vrai - Le tri devra être effectué dans l'ordre croissant, FAUX - dans l'ordre
décroissant).
"""

def TriTableau(F, T, N) :
    for i in range(0, N-1) :
        pos=i
        for j in range(i+1, N) :
            if (F) :
                if (T[j]<T[pos]) : #F est Vrai ==> l'ordre croissant
                    pos=j
            else:
                if (T[j]>T[pos]) :
                    pos=j
        temp=T[pos]
        T[pos]=T[i]
        T[i]=temp

T=[14,3, 1, 25, 24, 55, 54, 78, 100]
print(len(T))
print(T)
TriTableau(True,T, len(T))
print(T)

```

Exercice 5 : Tableau de deux dimensions

Soit un tableau T à deux dimensions (N et M) de valeurs numériques.

1. Écrire une procédure qui saisit les éléments de la matrice
2. Écrire une procédure qui affiche la matrice
3. Écrire une fonction qui retourne la plus grande valeur au sein de ce tableau.
4. Écrire le programme principal

Solution :

1. Écrire une procédure qui saisit les éléments de la matrice

Procédure SaisieMatrice(n, m: entier, tableau A)

Variable i, j : Entier

Début

 Pour i=0 à N-1 faire

 Pour j=0 à M-1 faire

 Ecrire ("Entrer les éléments de la ligne", i+1,"et la colonne", j+1)

 Lire(A[i][j])

 Fin pour

 Fin pour

Fin

```

import numpy as np
#1. Écrire une procédure qui saisit les éléments de la matrice
def SaisieMatrice(NL,NC, A):
    for i in range(NL) :
        for j in range(NC) :
            print("M[" ,i, "][",j, "]= ",end="")
            A[i][j]=int(input())
    return A

```


2. Écrire une procédure qui affiche la matrice

```
Procédure AfficheMatrice (N, M : Entier, A [] [] : Entier)
  Variable i, j : Entier
  Début
    Pour i=0 à N-1 faire
      Pour j=0 à M-1 faire
        Ecrire(A[i][j], " ")
      FinPour
      Ecrire ("\n")
    Fin pour
  Fin
```

```
#2. Écrire une procédure qui affiche la matrice
def AfficherMat(L,C, T) :
  for i in range(L) :
    for j in range(C) :
      print(T[i][j], end=" ")
    print(" ")
```

3. Écrire une fonction qui retourne la plus grande valeur au sein de ce tableau.

```
Procédure MaxMatrice (N, M : Entier, A [] [] : Entier)
  Variable i, j, i_Max, j_Max, Max : Entier
  Début
    Max=A[i][j]
    i_Max=0
    j_Max=0
    Pour i=0 à N-1 faire
      Pour j=0 à M-1 faire
        Si (A[i][j]>Max) alors
          Max A[i][j]
          i_Max= i
          j_Max=j
        Finsi
      FinPour
    Fin pour
    Ecrire ("La plus grande valeur du tableau est :", Max, "et sa position est :",
i_Max+1, j_Max+1)
  Fin
```

```
#3. Écrire une procédure qui affiche la plus grande valeur et sa position
def MaxMatrice(NL, NC, A):
  Max=A[0][0]
  i_Max=0
  j_Max=0
  for i in range(NL) :
    for j in range(NC) :
      if(A[i][j]>Max):
        Max=A[i][j]
        i_Max=i
        j_Max=j
  print ("La plus grande valeur du tableau est :", Max, "et sa position est :", i_Max+1, j_Max+1)
```

4. Écrire le programme principal

Algorithme Matrice

Variable tableau A[20][20] : réel

N,M,P;entier

Début

Ecrire("Entrer le nombre de ligne de la matrice A")

lire(N)

Ecrire("Entrer le nombre de colonne de la matrice A")

lire(P)

SaisieMatrice (N,P,A)

Ecrire ("La matrice A est : ")

AfficheMatrice(N,P,A)

MaxMatrice(N,P,A)

Fin

```
#4. Écrire le programme principal
NL=int(input("Donnez Le nombre de lignes :"))
NC=int(input("Donnez Le nombre de colonnes :"))
M=np.zeros((NL,NC))
M=SaisieMatrice(NL,NC, M)
print(M)
AfficherMat(NL,NC,M)
MaxMatrice(NL, NC, M)
```